IV International Conference on Particle-based Methods – Fundamentals and Applications
PARTICLES 2015
E. Oñate, M. Bischoff, D.R.J. Owen, P. Wriggers & T. Zohdi (Eds)

IMPROVING ACCURACY IN PARTICLE METHODS USING NULL SPACES AND FILTERS

Chris Gritton[1], Martin Berzins[2], Robert M Kirby[3]

SCI Institute, University of Utah, Salt Lake City, USA
[1] cgritton@sci.utah.edu [2] mb@sci.utah.edu [3] kirby@sci.utah.edu

Key words: Particle in Cell Method, Ringing Instability, Singular Value Decomposition, Material Point Method

Abstract. While particle-in-cell type methods, such as MPM, have been very successful in providing solutions to many challenging problems there are some important issues that remain to be resolved with regard to their analysis. One such challenge relates to the difference in dimensionality between the particles and the grid points to which they are mapped. There exists a non-trivial null space of the linear operator that maps particles values onto nodal values. In other words, there are non-zero particle values values that when mapped to the nodes are zero there. Given positive mapping weights such null space values are oscillatory in nature. The null space may be viewed as a more general form of the ringing instability identified by Brackbill for PIC methods. It will be shown that it is possible to remove these null-space values from the solution and so to improve the accuracy of PIC methods, using a matrix SVD approach. The expense of doing this is prohibitive for real problems and so a local method is developed for doing this.

1 Introduction to Particle in Cell Methods

Particle-in-cell, PIC, methods have been in use and development since the mid 1950's. The original PIC method was developed by F.H. Harlow as a hydrodynamics code [7] to handle fluid dynamics problems that involved large slips and distortions. The PIC method combines both Lagrangian and Eulerian components. The Lagrangian component involved particles that could be advected over the given domain and the Eulerian component involved a grid that would be used to perform calculations. While successful at achieving its original intent the original PIC was shown in the work done in the 1970s and 1980s [8, 3] to be subject to errors that are introduced into PIC calculations due to an aliasing affect caused by a mismatch in the degrees of freedom at the cell compared to the degrees of freedom at the particles. Brackbill [3] stated that, "Because the number of particles is finite, the number of Fourier modes is also finite. Thus, when there are n particles in each cell, there are n times as many Fourier modes as there are grid points."

When values are mapped from nodes to particles the lack of resolution at the nodes compared to resolution at the particles can cause an aliasing error. Again to quote Brackbill, "Aliases occur because all Fourier modes with wavelengths shorter than the grid spacing are indistinguishable at the grid points." [3].

The fluid implicit particle, FLIP, [4] method is a modification to the original PIC method that is applied to fluid dynamics problems. Like the PIC methods that have been used in plasma simulations, the FLIP method also has material properties that are carried with the Lagrangian particles. One major difference though from other PIC methods is that in most PIC methods velocities are calculated on the grid and then interpolated to the particles. In FLIP the change in velocity is calculated at the grid level and then interpolated to the particles where the particle velocity is then updated. By doing this there is a reduction in the numerical diffusion or viscosity due to interpolation. A derivative of the FLIP method is the material point method, MPM, [11] which extends the FLIP method to solid mechanics problems. Despite these and many other developments in particle methods [2, 1, 10, 12] the understanding of this phenomenon has not increased greatly. The intention here is to shed some light upon this issue.

2 The PIC Method Applied to Gas Dynamics

In considering the ringing instability Brackbill [3] started with simple gas dynamics linearized equations

$$\frac{\partial \rho_1}{\partial t} + u_0 \frac{\partial \rho_1}{\partial x} + \rho_0 \frac{\partial u_1}{\partial x} = 0 \tag{1}$$

$$\rho_0 \left[\frac{\partial u_1}{\partial t} + u_0 \frac{\partial u_1}{\partial x} \right] + \frac{\partial p_1}{\partial x} = 0, \tag{2}$$

where u_0 and ρ_0 are constants representing the constant flow and the initial uniform density, respectively, and u_1 and ρ_1 are the unknown velocity and density. Substituting $p_1 = c^2 \rho_1$ where c is the wave speed and rewriting these equations in terms of their material derivative,

$$\frac{D(.)}{Dt} = \frac{\partial(.)}{\partial t} + u_0 \frac{\partial(.)}{\partial x}, \tag{3}$$

After setting $\rho_0 = 1$, we get the following set of coupled equations,

$$\frac{D\rho_1}{Dt} + \frac{\partial u_1}{\partial x} = 0, \tag{4}$$

$$\frac{Du_1}{Dt} + c^2 \frac{\partial \rho_1}{\partial x} = 0. \tag{5}$$

Applying the $\frac{D(.)}{Dt}$ operator to equation 4 and the $\frac{\partial(.)}{\partial x}$ operator to equation 5 gives, after some simple manipulation, the form of the wave equation on a grid moving with the

velocity u_0 as,

$$\frac{D^2\rho_1}{Dt^2} = c^2\frac{\partial^2\rho_1}{\partial x^2}. \tag{6}$$

If our initial conditions are $\rho_1(x,0) = f(x)$ and $\frac{\partial\rho_1(x,0)}{\partial t} = 0$ and the boundary conditions are periodic then the solution [9] in the fixed frame is the modified d'Alembert's formula,

$$\rho_1(x,t) = \frac{1}{2}(f(x-(c-u_0)t) + f(x+(c-u_0)t)). \tag{7}$$

2.1 Mapping Matrix

In particle methods, it is necessary to map values from the particles to the nodes [11, 2]. For example, one mapping of values g_p, where $g_p = g(x_p)$ are the values at particles, to the nodes can be written as,

$$g_i = \frac{\sum_p g_p\phi_{ip}}{\sum_p \phi_{ip}}, \tag{8}$$

$$= \sum_p S_{ip}g_p. \tag{9}$$

where ϕ_{ip} is the linear basis function $\phi_i(x)$ with value 1 at node x_i and zero at other nodes evaluated at the particle point x_p. This mapping from particles to node can also be expressed in terms of a system-wide matrix

$$\mathbf{g}_i = \mathbf{S}_{ip}\mathbf{g}_p, \tag{10}$$

where $\mathbf{g}_i$ contains the mapped values at the nodes and $\mathbf{g}_p$ are the values at the particles.

2.2 PIC Method Description

One PIC type approach maps particle values to the nodes and then calculates the gradient by taking the gradient of the interpolating function. Using the piecewise linear basis functions, $\phi_i(x)$, the data values at the nodes the linear approximation to the function $g(x)$ can be defined as follows,

$$g_l(x) = \sum_i g_i\phi_i(x). \tag{11}$$

The gradient of the function $g(x)$ can be defined as,

$$\nabla g_l(x) = \sum_i g_i\nabla\phi_i(x). \tag{12}$$

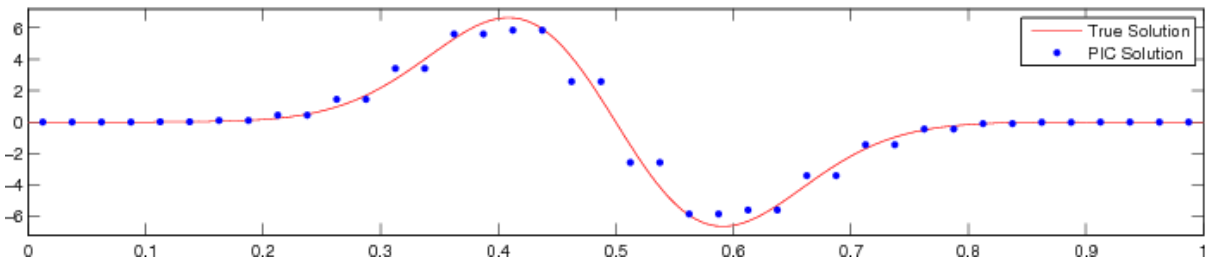

Figure 1: Computed versus true solution for $\frac{\partial g(x)}{\partial x}$

For a piecewise linear function, the gradient of $\phi_i(x)$ is defined as,

$$\nabla\phi_i(x) = \begin{cases} \frac{1}{h} & \text{if } x_{i-1} \leq x < x_i \\ \frac{-1}{h} & \text{if } x_i \leq x < x_{i+h}. \end{cases} \tag{13}$$

When using a piecewise linear basis function, the gradient is piecewise constant across each interval. Figure 1 shows the comparison between the computed gradients at the particles and the true solution given by $g(x) = e^{-60(x-0.5)^2}$ and so $\frac{\partial g(x)}{\partial x} = -120(x-0.5)e^{-60(x-0.5)^2}$. The pairing of point values is due to the piecewise constant derivative of the linear basis functions.

3 Null Space of the Mapping Matrix

The matrix $\mathbf{S}_{ip}$ is rectangular and may have a nontrivial nullspace. For example, let $\mathbf{c}$ be a vector in $\mathbb{R}^n$. If $\mathbf{S}_{ip}\mathbf{c} = \mathbf{0}$, then we say that $\mathbf{c}$ is in the nullspace of $\mathbf{S}_{ip}$. We can determine the nullspace of $\mathbf{S}_{ip}$ by making use of its singular value decomposition, SVD [5]. Taking the SVD of $\mathbf{S}_{ip}$ gives the following decomposition,

$$\mathbf{S}_{ip} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \tag{14}$$

where $\mathbf{U}$ has dimension m by m, $\mathbf{\Sigma}$ is m by n, and $\mathbf{V}$ is n by n. The matrices $\mathbf{U}$ and $\mathbf{V}$ are unitary, meaning that the columns are orthonormal [13]. In other words, if $\mathbf{u}_i$ and $\mathbf{u}_j$ are columns of the matrix U, then,

$$\mathbf{u}_i^T\mathbf{u}_j = \delta_{ij}, \tag{15}$$

where the superscript T is the transpose of the vector and δ_{ij} is the Kroenecker delta. The columns of $\mathbf{U}$ and $\mathbf{V}$ are orthogonal, linearly independent and span the spaces $\mathbb{R}^m$ and $\mathbb{R}^n$ respectively, [13]. Any vector $\mathbf{a} \in \mathbb{R}^m$ can be expressed as a linear combination of the columns of $\mathbf{U}$.

$$\mathbf{a} = c_1 \begin{bmatrix} u_1 \end{bmatrix} + \cdots + c_m \begin{bmatrix} u_m \end{bmatrix} \tag{16}$$

where $c_1, \ldots, c_m$ are constants. The matrix $\mathbf{\Sigma}$ is an m by n matrix of the form

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1 & 0 & \ldots & 0 & 0 & \ldots & 0 \\ 0 & \sigma_2 & \ldots & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & 0 \\ 0 & 0 & \ldots & \sigma_r & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ldots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 & 0 & \ldots & 0 \end{bmatrix} \tag{17}$$

in which the columns 1 to r are zero apart from their diagonal elements which contain the nonzero singular values $\sigma_1, \ldots, \sigma_r$ and the columns $r+1$ to n are zero, [5, 13]. Taking the matrix product of $\mathbf{\Sigma}$ and $\mathbf{V}^T$, gives,

$$\mathbf{\Sigma V}^T = \left(\mathbf{V\Sigma}^T\right)^T = \left[\begin{array}{c|c|c|c|c|c} & & & & & \\ \sigma_1\mathbf{v}_1 & \ldots & \sigma_r\mathbf{v}_r & 0 * \mathbf{v}_{r+1} & \ldots & 0 * \mathbf{v}_n \\ & & & & & \end{array}\right]^T . \tag{18}$$

Consequently the column vectors $\mathbf{v}_{r+1}$ to $\mathbf{v}_n$ span the nullspace of $\mathbf{\Sigma}$, which in turn means that they span the nullspace of $\mathbf{S_{ip}}$. From equation 16, any vector can be decomposed into its orthogonal components [13]. Since the columns of $\mathbf{V}$ are orthogonal, they form a basis for $\mathbb{R}^n$, which means that any vector $b \in \mathbb{R}^n$ can be expressed as a linear combination of the columns of $\mathbf{V}$,

$$\mathbf{b} = c_1 \begin{bmatrix} \\ \mathbf{v}_1 \\ \\ \end{bmatrix} + \cdots + c_r \begin{bmatrix} \\ \mathbf{v}_r \\ \\ \end{bmatrix} + \underbrace{c_{r+1} \begin{bmatrix} \\ \mathbf{v}_{r+1} \\ \\ \end{bmatrix} + \cdots + c_m \begin{bmatrix} \\ \mathbf{v}_n \\ \\ \end{bmatrix}}_{null(S_{ip})} . \tag{19}$$

From this it can be seen that a portion of $\mathbf{b}$ is in the nullspace of $\mathbf{S}_{ip}$. Using the inner product [13] allows the components of a vector that are in the null space to be found. Given a vector $\mathbf{u}_p$ if we define the vector $\mathbf{r}_i$ as,

$$\mathbf{r}_i = \mathbf{u}_p - (\mathbf{v}_i^T \mathbf{u}_p)\mathbf{v}_i, \tag{20}$$

then as $(\mathbf{v}_i^T \mathbf{r}_i) = 0$ it follows that $\mathbf{r}_i$, has no component in the direction of $\mathbf{v}_i$. The vector $\mathbf{u}_p$ can now be expressed as linear combination of two vectors,

$$\mathbf{u}_p = (\mathbf{v}_i^T \mathbf{u}_p) \begin{bmatrix} \\ \mathbf{v}_i \\ \\ \end{bmatrix} + \begin{bmatrix} \\ \mathbf{r}_i \\ \\ \end{bmatrix} . \tag{21}$$

Repeating this process using the first r column vectors of V, and defining the vector $\mathbf{r}$ by,

$$\mathbf{r} = \mathbf{u}_p - \sum_{i=1}^{r} (\mathbf{v}_i^T \mathbf{u}_p)\mathbf{v}_i, \tag{22}$$

gives a vector, $\mathbf{r}$, that is the sum of the components of $\mathbf{u}_p$ that lie entirely in the nullspace of $\mathbf{S}_{ip}$. The filtered form of $\mathbf{u}_p$ with no nullspace component is denoted here by $\mathbf{u}_p^F$ and given by

$$\mathbf{u}_p^F = \sum_{i=1}^{r} (\mathbf{v}_i^T \mathbf{u}_p)\mathbf{v}_i. \tag{23}$$

This filtering operation can be described using a Matlab style of syntax in a function called $Filter()$ that takes as its parameters an n dimensional vector $\mathbf{u}_p$ and the matrix $\mathbf{S}_{ip}$, uses a function called $svd()$, which decomposes a matrix into $\mathbf{U}$, $\mathbf{\Sigma}$, and $\mathbf{V}$ and a function called $rank()$, which returns the number of singular values in a matrix. The function returns the filtered form of $\mathbf{u}_p^F$ in $\mathbf{u}_p$

```
function Filter(up, Sip)
    U, S, V = svd(Sip)    \\ S = singular values
    k = rank(S)
    r = 0
    for i = 1 to k
        r = r +  (up' * V(:,i))*V(:,i)
    end
    return r
```

The central idea of this paper is to use a filter such as this to remove numerical noise associated with the null space.

3.1 Removing the Nullspace

Equation 11 maps particle values to the nodes. In matrix form, this is,

$$\mathbf{g}_i = \mathbf{S}_{ip}\mathbf{g}_p. \tag{24}$$

At this point, the nullspace component of $\mathbf{g}_p$ has been removed by the nature of the mapping. It is at the next step in the computation, equation 12, that a nullspace component can be re-introduced. If we express the computed gradients at the particles, ∇g_p, in the vector form $\mathbf{dg}_p$ then we can decompose the vector into a nullspace and a non-nullspace component in the same way as above. Figure 2 show what the nullspace components look like. Once the nullspace component is known, it can be removed from the computed gradient, to get a smoothed version of $\mathbf{dg}$. Figure 3 show $\mathbf{dg}$ with the nullspace component removed.

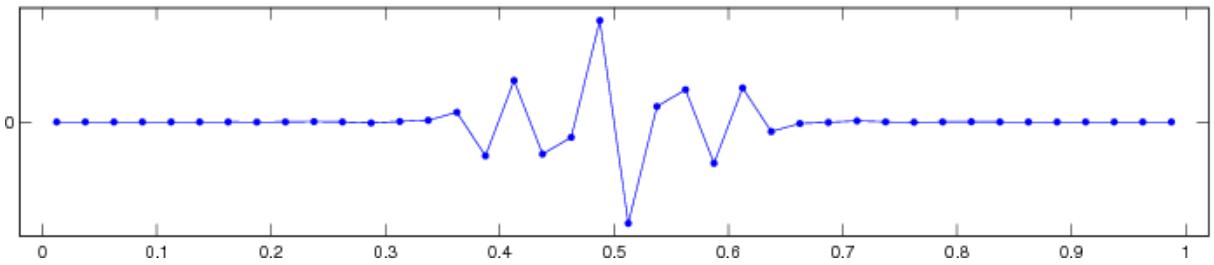

Figure 2: Nullspace of computed $\frac{\partial g(x)}{\partial x}$

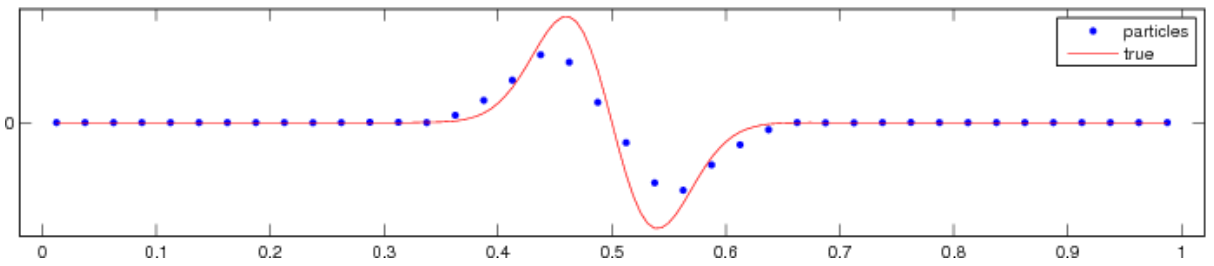

Figure 3: The results of $\frac{\partial g(x)}{\partial x}$ with the nullspace component removed

3.2 Nodalwise Noise Removal

Using singular value decomposition for the removal of nullspace noise works well for small one-dimensional problems, but it does not scale well when running a multidimensional simulation across multiple cores. A second issue is that the computational complexity of generating the matrix $\mathbf{V}$ with a singular value decomposition is $O(m^2n + n^3)$ [5]. In order to have a method for removing the nullspace noise that scales well across hundreds of cores, we need a new approach that works locally across just a few nodes and not the entire set of nodes.

3.2.1 Local Method

This method takes a different approach then the SVD method to removing the nullspace components. The key idea in the local method is to use the already mentioned fact that any vector, $\mathbf{a} \in \mathbb{R}^n$, can be decomposed into a nullspace and non-nullspace component, so when the matrix $\mathbf{S}_{ip}$ is applied to the vector $\mathbf{a}$, the nullspace portion of $\mathbf{a}$ is removed. In our case, the gradient at the particles is computed by first mapping particle values to the nodes, using equation 9. and then computing the gradients at the particles by using the gradient of the the interpolating function that interpolates values from nodes to particles, equation 12. When this happens, a nullspace component is introduced by this gradient calculation. Taking advantage of the observation made earlier, if the newly computed

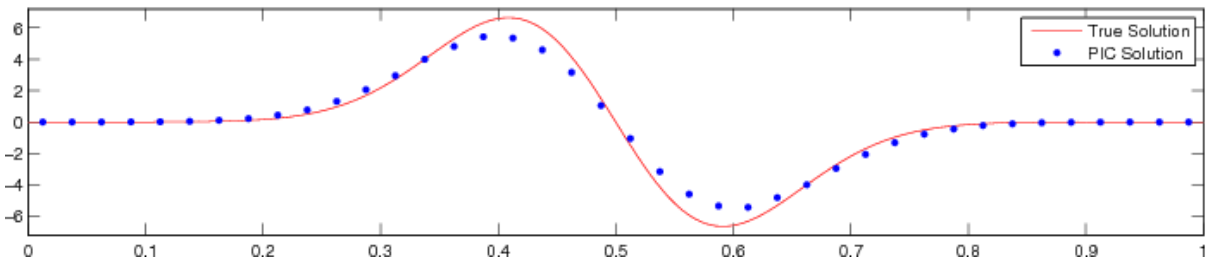

Figure 4: Comparison between the true solution and the computed solution using the local method as a filter for $\frac{\partial g}{\partial x}$

gradient values are mapped back to the nodes,

$$\nabla g_i = \frac{\sum_p \phi_{ip} \nabla g_p}{\sum_p \phi_{ip}}, \tag{25}$$

or in matrix form,

$$\nabla \mathbf{g}_i = \mathbf{S}_{ip} \nabla \mathbf{g}_p, \tag{26}$$

then its nullspace component is removed. With gradient values mapped to the nodes, all that needs to be done now is to interpolate the values back to the particles,

$$\nabla g_p = \sum_i \nabla g_i \, \phi_{ip}. \tag{27}$$

While there is no nullspace component at the nodes, there is nothing to prevent nullspace noise from being introduced at the particle via the interpolation process. Furthermore, there is no longer a need to explicitly calculate the nullspace in order to smooth out the particle gradients. Figure 4 show a comparison between the true gradient and the calculated solution at the particles.

3.3 PIC Formulations for Compressible Flow

The approach used here is to map only the particle values for ρ to the nodes nodes, to compute the gradient of ρ, and update the particle values of u. The updated particle values for u are then mapped to the nodes. The gradients of u are then calculated and are used to update the particle values of ρ. Finally, the particles are then moved.

This appraoch has at its heart a stable nodal method [6]. We use Formulation 2 as defined by [6] in its vector form (p36-37) and its filtered form as defined above using the Filter function.

Formulation 2	Formulation 2 Filtered
1. $u_i^t = \sum_{p=1}^{np} S_{ip} u_p^t$	1. $u_i^t = \sum_{p=1}^{np} S_{ip} u_p^t$
2. $\rho_p^{t+1} = \rho_p^t + c\frac{dt}{h}(u_{i+1}^t - u_i^t)$	2. $du = Filter((u_{i+1}^t - u_i^t), S_{ip})$
3. $\rho_i^{t+1} = \sum_{p=1}^{np} S_{ip} \rho_p^{t+1}$	3. $\rho_p^{t+1} = \rho_p^t + c\frac{dt}{h}(du)$
4. $u_p^{t+1} = u_p^t + c\frac{dt}{h}(\rho_{i+1}^{t+1} - \rho_i^{t+1})$	4. $\rho_i^{t+1} = \sum_{p=1}^{np} S_{ip} \rho_p^{t+1}$
5. $x_p^{t+1} = x_p^t + vdt$	5. $d\rho = Filter((\rho_{i+1}^{t+1} - \rho_i^{t+1}), S_{ip})$
	6. $u_p^{t+1} = u_p^t + c\frac{dt}{h}(d\rho)$
	7. $x_p^{t+1} = x_p^t + vdt$

Note that to avoid confusion in notation, the variable v is used instead the variable u_0 that is found in equations 1 and 2 to represent the particle velocity. The nullspace filter is applied to this model by applying the filter function to the updates to u and ρ. In this case we first need to construct the matrix $\mathbf{S}_{ip}$. It is important to remember that this needs to be done after each iteration because the mapping is dependent on particle position. Two options are available when filtering. The first is to use the full SVD filter while the second is to use the local filter. It is now possible to look at how the nullspace noise can be filtered out during each computation cycle. As discussed above, nullspace noise can be injected any time a value goes from node to particle. When the interpolation is linear, the noise tends to be minimal, but in the cases when the piecewise constant gradient calculated using nodal values is mapped to a particle, then there tends to be more nullspace noise. There are two points in Formulation 2 when this occurs. The first is when the gradient of u_i is calculated, and the second is when the gradient at ρ_i is calculated.

3.4 Comparison of Filtered and Nonfiltered Formulations

Figures 5, 6, and 7 are snapshots from a simulation that was run comparing Formulation 2 without the nullspace filter and Formulation 2 with the nullspace filter. As can be seen by Figure 6 at time step 700, the method without the nullspace filter is showing the results of the numerical nullspace noise. On the other hand, it is shown in Figure 7 that the method with the nullspace filter is still stable at 1000 steps even though the original method is clearly unstable. One interesting feature of the two filters is that if the calculation is continued out too 5000 steps the solution obtained with the local filter remains stable but the SVD filtered solution does finally become unstable. The two solutions are shown in Figure 8 We have also experimented with the use of this filter with the Material Point Method. Numerical calculations show that the MPM method

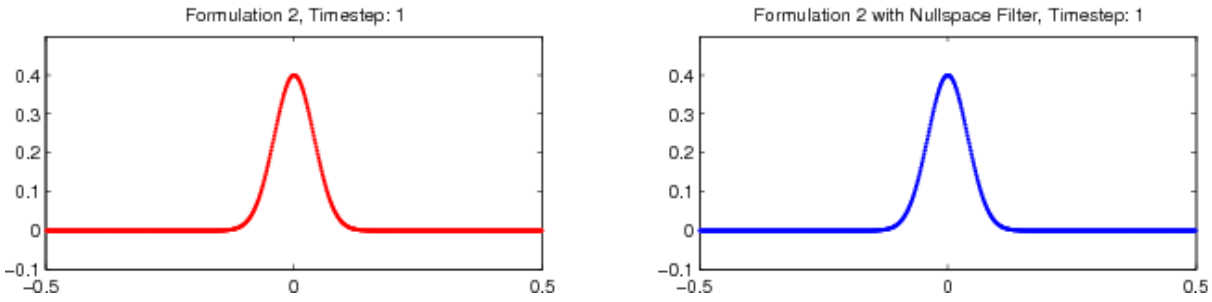

Figure 5: Initial time step

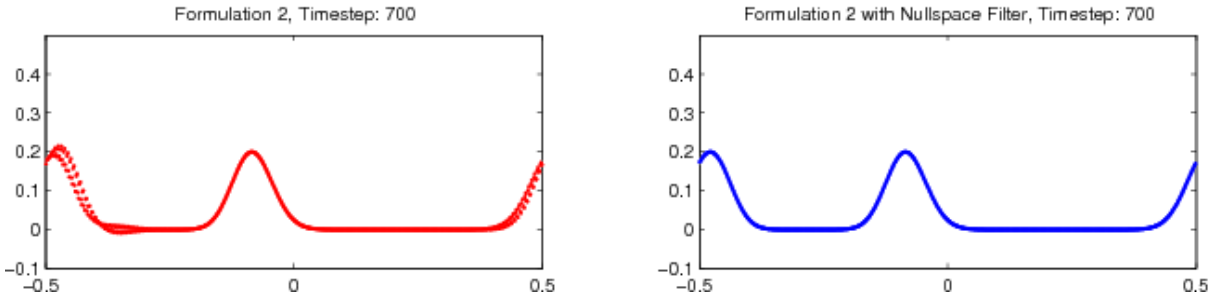

Figure 6: Time step 700

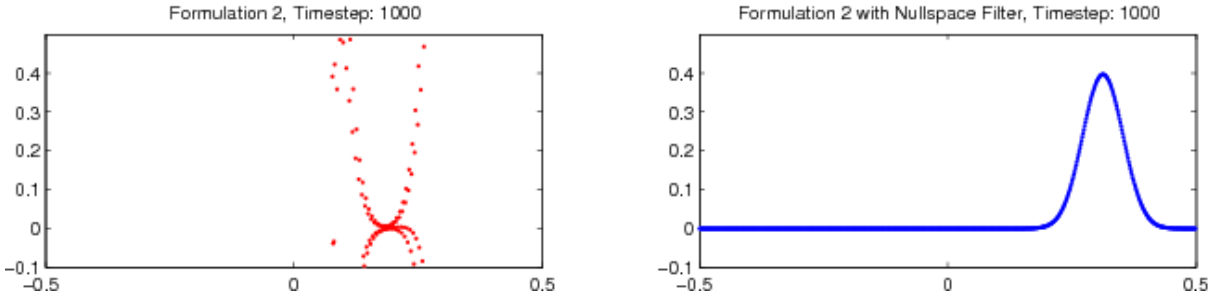

Figure 7: Time step 1000

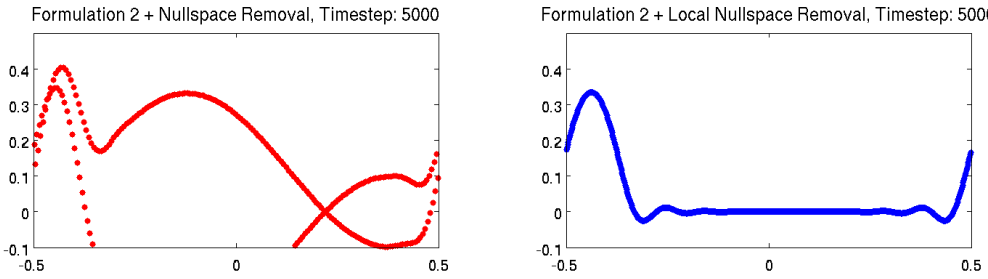

Figure 8: Time step 5000

has some important advantages over the PIC approach, [6]. In part these appear due to the different formula used by MPM in discretising the wave equation. Our observation is that it is the grid crossing error derived by [12] and discussed by [10] that may dominate the error in cases such as those considered above, but further work is required.

4 Conclusion

The removal of the nullspace error from the PIC method used has resulted in much improved stability. While the original method uses an expensive SVD decomposition, a local method that is much less expensive has also been proposed. This local method also seems to work well. These ideas have also been applied to the MPM method in work that will be reported elsewhere.

Acknowledgement: Research was primarily sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-12-2-0023. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.

REFERENCES

[1] S. Bardenhagen, *Energy conservation error in the material point method for solid mechanics*, Journal of Computational Physics, 180 (2002), pp. 383 – 403.

[2] S. Bardenhagen and E. Kober, *The generalized interpolation material point method*, Computer Modeling in Engineering and Science, 5 (2004), pp. 477 – 495.

[3] J. Brackbill, *The ringing instability in particle-in-cell calculations of low-speed flow*, Journal of Computational Physics, 75 (1988), pp. 469 – 492.

[4] J. Brackbill, D. Kothe, and H. Ruppel, *Flip: A low-dissipation, particle-in-cell method for fluid flow*, Computer Physics Communications, 48 (1988), pp. 25 – 38.

[5] G. H. Golub and C. F. V. Loan, *Matrix Computations*, The John Hopkins University Press, third ed., 1996.

[6] C.E. Gritton, *Ringing Instabilities in Particle Methods*, M.S.Thesis in Computational Engineering and Science, August 2014, School of Computing, University of Utah.

[7] F. H. Harlow*The particle-in-cell method for fluid dynamics*, Methods in Computational Physics, 3 (1964).

[8] A. Langdon, *Effects of the spatial grid in simulation plasmas*, Journal of Computational Physics, 6 (1970), pp. 247 – 267.

[9] J. D. Logan, *Applied Partial Differential Equations*, Springer-Verlag, New York, second ed., 2004.

[10] M. Steffen, R. M. Kirby, and M. Berzins, *Analysis and reduction of quadrature errors in the material point method (mpm)*, International Journal for Numerical Methods in Engineering, 76 (2008), pp. 922–948.

[11] D. Sulsky, Z. Chen, and H. Schreyer, *A particle method for history-dependent materials*, Computer Methods in Applied Mechanics and Engineering, 118 (1994), pp. 179 – 196.

[12] L.T. Tran and J. Kim and M. Berzins, *Solving Time-Dependent PDEs using the Material Point Method, A Case Study from Gas Dynamics*, International Journal for Numerical Methods in Fluids", 62,7,709–732",2009.

[13] L. N. Trefethen and I. David Bau, *Numerical Linear Algebra*, SIAM, 1997.